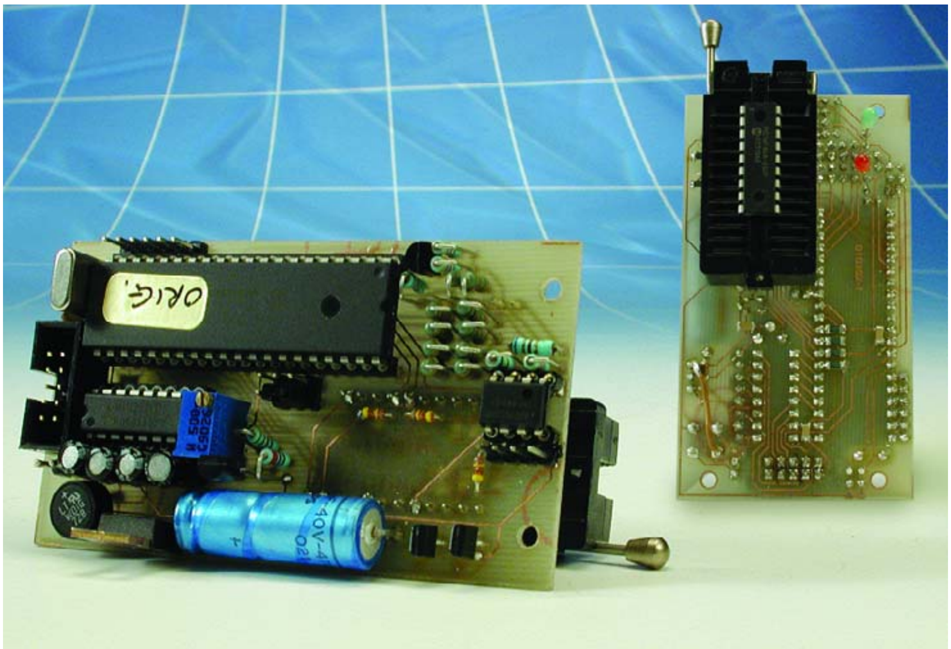
PICProg 2003

A versatile PIC production programmer

Design by R. Brauns

This programmer was specially designed for those of you requiring a cheap way to program PIC microcontrollers, without sacrificing useful features as compared with professional programmers on the market. The circuit described here is capable of handling more than 85 different types of PIC micro!



Suitable for nearly all types from these PIC series:

12C(E)xxx
12Fxxx
16Cxxx
16Fxxx

The popularity of PIC microcontrollers has seen a healthy increase these past few years, particularly when the first chips with Flash memory were introduced. Since the launch of the most popular member of the PIC family, the 16F84, dozens of successor types have made it to industrial as well as hobby use, not in the least thanks to projects and New Product releases published in *Elektor Electronics*. Of the more recent 'beasts' produced by Microchip, the PIC16F627/628 come to mind, or the PIC16F818/PIC16F819 which is a perfect replacement for the PIC16F84. These micros offer, among others, extra hardware for an RS232 link, a PWM (pulsewidth modulation) module and additional timers.

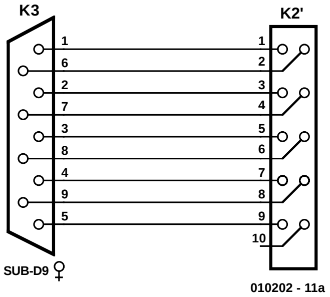
If you require a microcontroller for heavier tasks (for instance, one with an on-chip ADC) then take a look around in the 16F87x series it is well worth your effort.

Microchip has replaced as number of EPROM based PICs (like those in the 16C7x series) by Flashable versions. A number of Flash replacements are now also available for popular PICs from the 12Cxx series. For example, have a look at the 12F629 (with a unit price below £1) or its 14-pin sibling the 16F630.

Extensive information on all PIC microcontrollers may be found on the manufacturer's website: www.microchip.com.
The website also has a wide variety of 'tools of the PIC trade' including datasheets, an assembler and development platforms for DOS as well as Windows.

As already intimated, this project is not just another standard programmer of which dozens may be found on the Internet and competing magazines. Valuable as they are, such programmers either have severe limitations in the number of PIC types they can handle, or they simply do not work properly for some types proudly included in the 'supported devices' list. That is why we searched for an inexpensive solution to make a 'production programmer' (a term used by Microchip). The cost of our DIY programmer is about half that of comparable PIC programmers on the market, and the general design and quality is well beyond what may be expected from any of our competing magazines.

The programmer is connected to the RS232 port on a host PC, using a 1-to-1 (i.e., non-crossed) RS232 cable. All you need in the way of a power supply is a commercial 15-VAC mains adapter capable of supplying 300 mA or more.



The project software has been implemented for Windows 95/98/NT/ME/2000/XP. Thanks to the Flash memory, the firmware in the controller on the programmer board is easily updated via the serial port, allowing you to install updates when new PICs are being introduced like, for example, the latest PIC18xxx types and the rapidly growing Flash family.

Finally, we hope that the publication of this programmer acts as a stimulant for you to consider using other PICs than the run-of-the-mill 16F84.

Circuit diagram

Despite the versatility of the PIC-Pro 2003, its schematic is relatively simple, see **Figure 1**. The heart of the circuit is formed by IC6, a ready-programmed PIC16F874-20P. It has been programmed to look after the programmer signal timing, the voltages at the programmer socket pins and, of course, the communication with the host PC.

The connection with a (free) serial port on the PC is established via K2. The data speed on the serial link is set to 115.2 kbits/s. RS232 signals are converted to TTL swing, and the other way around, by IC4. Bi-directional communication is made as reliable as possible by the use of the CTS and RTS handshaking lines on the RS232 interface. These lines force the system to 'idle' until received bytes have been processed. Of course, this handshaking will tend to slow down the communication a little. Fortunately, that is not a problem because no PIC contains more than 8 kbytes of

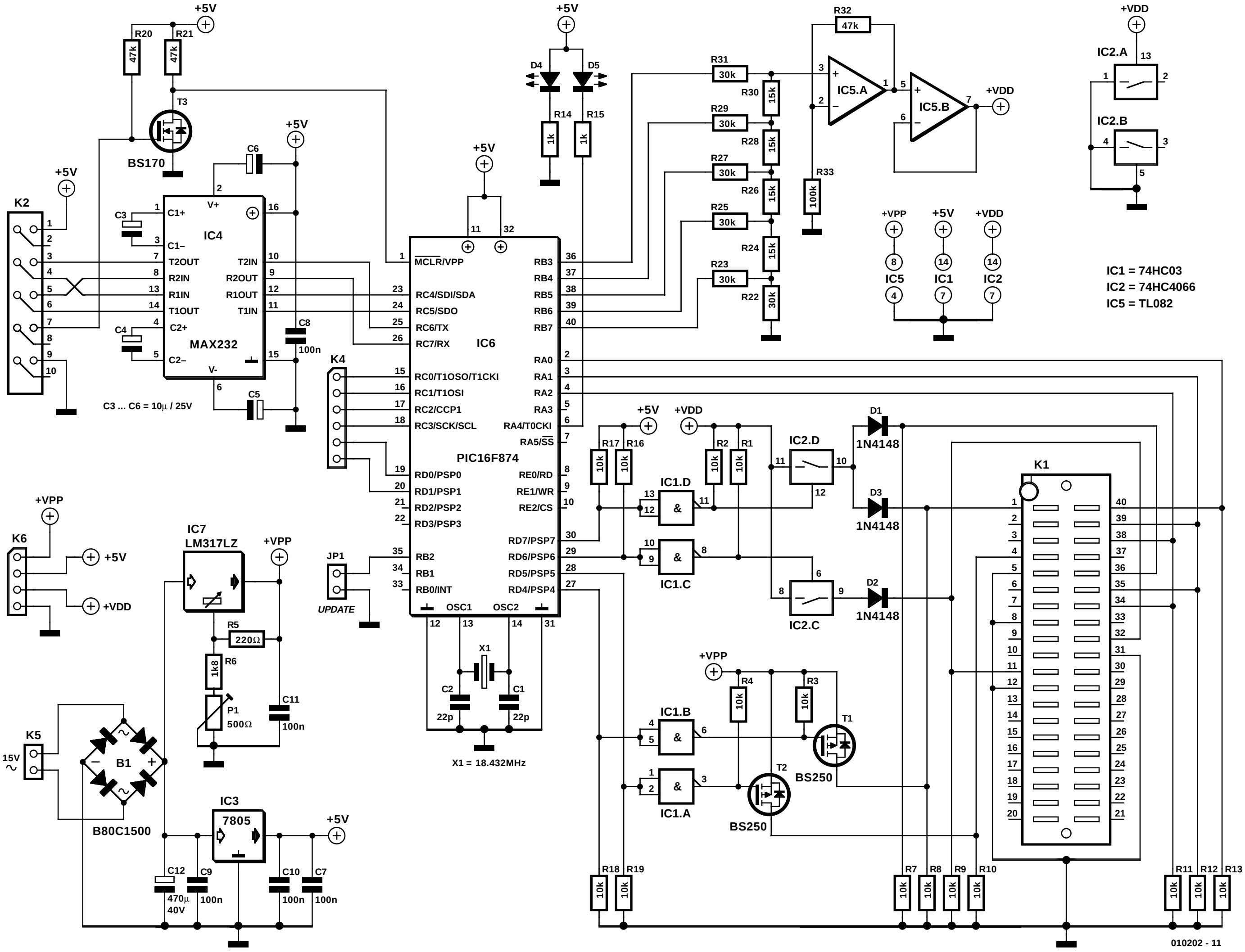


Figure 1. Despite the extremely versatile character of the PICProg 2003, its complexity remains with reason.

COMPONENTS LIST

Resistors:
R1-R4,R7-R13,R16-R19 = 10kΩ SMD
R5 = 220Ω
R6 = 1kΩ8
R14,R15 = 1kΩ
R20,R21,R32 = 47kΩ
R22,R23,R25,R27,R29,R31 = 30kΩ0 1%
R24,R26,R28,R30 = 15kΩ0 1%
R33 = 100kΩ
P1 = 500Ω preset, vertical, multiturn

Capacitors:
C1,C2 = 22pF
C3-C6 = 10µF 25V radial
C7-C11 = 100nF SMD
C12 = 470µF 40V axial

Semiconductors:
D1,D2,D3 = 1N4148
D4 = LED, green, 3mm, low-current
D5 = LED, red, 3mm, low-current
T1,T2 = BS250
T3 = BS170
IC1 = 74HC03 SMD
IC2 = 74HC4066 SMD
IC3 = 7805
IC4 = MAX232
IC5 = TL082CP
IC6 = PIC16F874-20/P (programmed, order code **010202-41**)
IC7 = LM317LZ
B1 = B80C1500 (round case) (80V piv, 1.5A)

Miscellaneous:
K1 = 40-way ZIF programming socket, plus a 10-way and a 12-way SIL contact strip
K2 = 10-way boxheader
K3 = 9-way sub-D socket (female), with IDC connector (not on PCB!)
K4 = 6-way SIL pinheader
K5 = 2-way pinheader
K6 = 4-way SIL pinheader
JP1 = 2-way SIL pinheader with jumper
X1 = 18.432MHz quartz crystal
10-way IDC connector
Length of 10-way flatcable
Case, size 120 x 60 x 40 mm, e.g., Velleman # G410
Connector for mains adapter, chassis mount
PCB, order code **010202-1**
Disk, Windows software, order code **010202-11** or Free Download
Runtime files, file number **010202-12**, Free Download

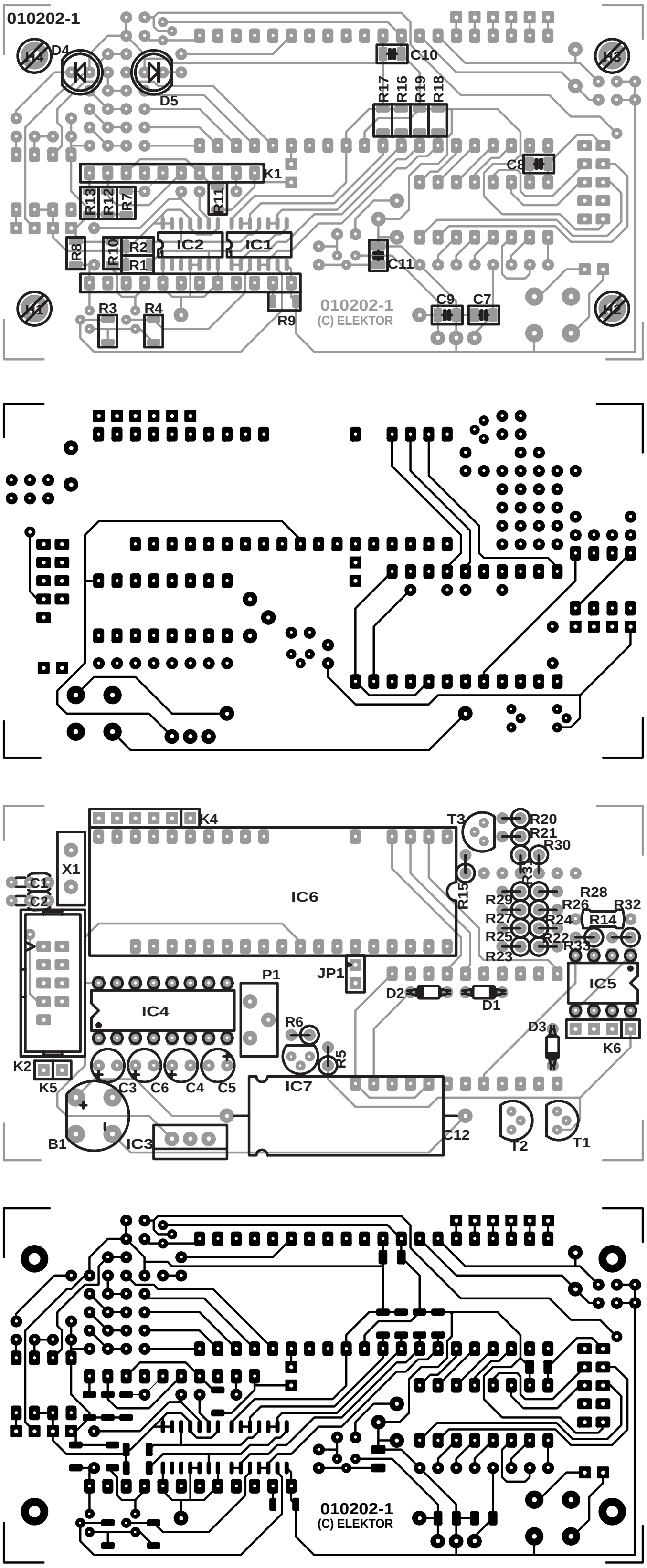


Figure 2. The double-sided board holds parts at both sides. One of these is the programming socket, K1 (board available ready-made).

program memory.

Components R22-R31 and IC5a/b together form a cheap and simple D-A converter with a buffered output. This DAC is used to generate the variable supply voltage needed for the PIC to be programmed. This voltage has to be variable between 2 V and 6 V in 0,25 V steps. A simple R-2R resistor

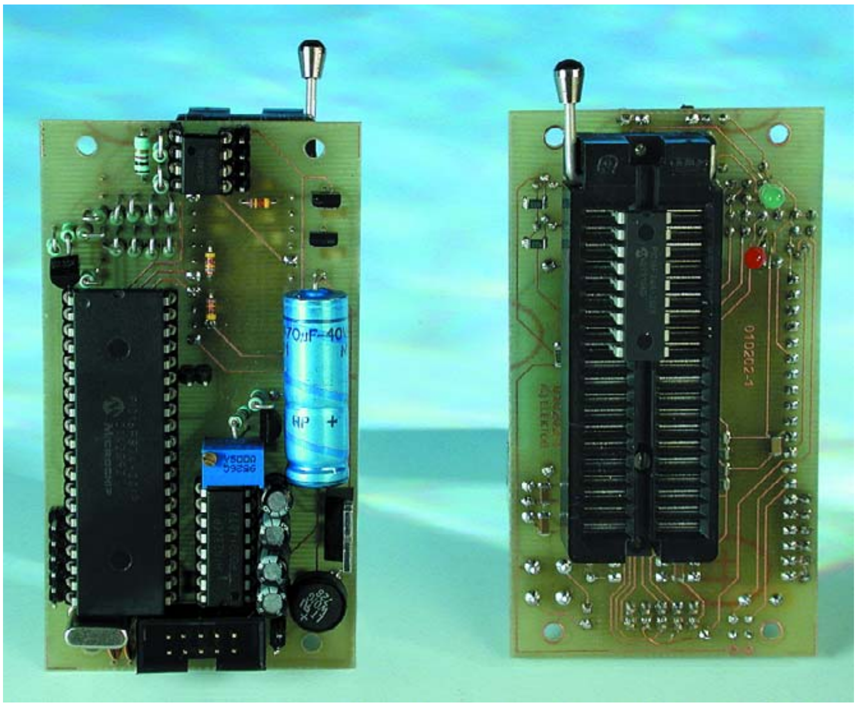


Figure 3. Top side and bottom side of the finished board (laboratory prototype).

ladder fits the bill perfectly, provided we use resistors with a tolerance not exceeding 1%.

The variable voltage is applied to the relevant pins of the programming socket K1 by way of electronic switches IC2c and IC2d. Once on the socket pin, the programmable voltage acts as V_{DD} for the PIC chip. Diode D3 protects switch IC2d against too high voltages at its output. After all, excessive levels may occur when T1 starts to conduct and 13 V arrives on pin 1 of K1. The other diodes are included to ensure a uniform voltage drop at all pins.

As already mentioned, a voltage of about 13 V is applied to the programming socket via T1 and T2. If the MCLR pin of the controller to be programmed is pulled to 13 V, the chip will enter programming mode. MOSFETs have been chosen in this design because of their low voltage drop across the conducting drain-source junction. The FETs as well as IC2 operate at different supply voltages with respect to 5 V. That is why the gates in IC1 provide a level adaptation between these components and controller IC6.

Four resistors, R16-R19, connected to GND or V_{cc} (5 V) have been included at the inputs of the gates in IC1. These resistors make sure no voltage remains at the programming socket pins when IC6 is reset, since as at that instant all pins of IC3 are configured as inputs. Likewise, R7-R13 ensure that unused inputs or pins configured as an input are not allowed to 'float'.

Because a voltage of 13 V is needed in the programmer, it is not possible to resort to the use of a standard mains adapter of the DC type (a.k.a. 'battery eliminator'). Most mains adapters have a maximum output voltage of 12 V, and versions with a higher output voltage are expensive, hard to get, or both. We could have designed a DC/DC converter into the circuit, but that would have meant a considerable cost increase for the project. The problem has been solved by the use of a relatively inexpensive 15-VAC mans adapter. It should be noted, however, that most, if not all, DC adapters of the low-cost type supply much higher output voltages than specified (particularly at light loads) so do check with a voltmeter, you may have one that's suitable for this circuit.

Using B1 and C12 the adapter voltage is rectified and smoothed. The raw voltage is then applied to two regulators: IC3 for the 5-V rail and IC7 for the 13-V programming voltage. The 13-V level may be accurately adjusted with the aid of P1.

Around RS232 connector K2 we find a small circuit comprising two resistors and a BS170 FET. These parts ensure that IC6 is automatically reset when no PC is connected. Also, a reset is generated at the start of every command (for example, reading the program memory). This is done to make absolutely sure the PICPro and the PC keep communicating even if an error occurs owing to external circumstances.

Connectors K4 and K6 are provided for future extensions. The author has plans to develop a module for ISP (in circuit programming), which he hopes to be able to mount under the programmer board.

The executable code inside IC6 may be updated in a simple manner using a special Windows program. This requires jumper JP1 to be fitted first. Normally, this jumper is omitted for security reasons!

The green LED, D4, indicates that the circuit receives its supply voltage. The red LED, D5, lights when voltages are being applied to the programming socket. **In this condition, ICs should not be inserted into the ZIF socket or removed from it**.

Construction

All components are accommodated on a double-sided, through-plated printed circuit board, of which the artwork is shown in **Figure 2**.

If you decide to use IC sockets for positions IC4, IC5 and IC6 than go in sockets with turned pins. Most resistors are mounted vertically. Like the printed circuit board, the ready-programmed PIC may be ordered through the *Elektor Electronics* Readers services, the order code is **010102-41**.

Two SMD integrated circuits, a number of SMD passive components and programming socket K1 are mounted **at the solder side of the board**. The SMD parts have to be fitted accurately using a minimum amount of solder and a soldering iron with a fine tip.

Having fitted all parts on to the board, it is time for a thorough visual inspection of your work so far. If everything appears to be in order, the PCB may be mounted in a suitable ABS case (see parts list). The RS232 connector and the mains adaptor socket are mounted on to the case and connected to K2 and K5 using short cables. The RS232 connection may take the form of a short piece of flatcable with press-on (IDC) connectors, provided you have the matching 9-way sub-D connector.

Before you solder the LED terminals in place, check their lengths to make sure the

LEDs can be seen through the two holes in the case.

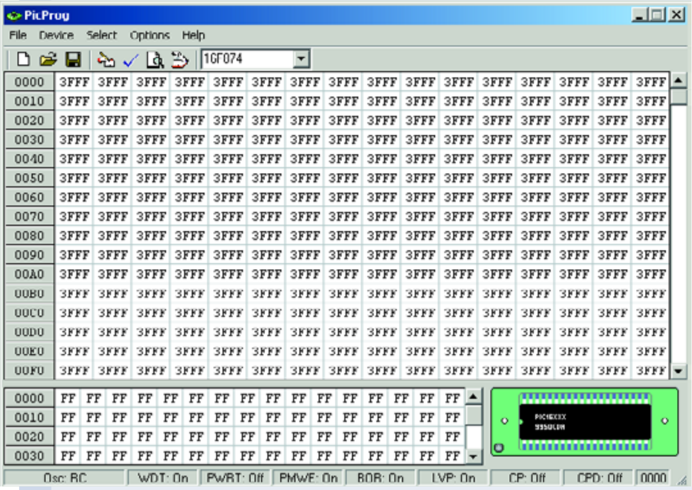
The programming socket is a 40-way ZIF (zero insertion force) type with wide slots so that narrow DIP ICs can be programmed also. At the solder side of the board, first solder a 10-way and a 12-way SIL socket strip in position K1. Next, insert the ZIF socket into these receptacles, making sure pin 1 of the ZIF socket is beside resistor R8. If the socket does not protrude

Two Windows programs

The installation of the two Windows utilities for the PICProg 2003 programming system is rather simple: copy the folders **PicProg** and **Update**, complete with their contents, to a suitable directory on the hard disk. Next, create desktop shortcuts to the programs **Pcprog.exe** and **Update.exe**. If error reports pop up, these are probably caused by a number of missing Visual Basic runtime modules. If necessary, you may download them from the Elektor Electronics website, the file number being **010102-12**. Next, copy the modules into the Windows/System/ folder. Fortunately, the runtime modules are already present on most reasonably modern PCs.

Main program ‘PicProg’

The screenshot opposite shows the program PicProg in action. The menu options are briefly discussed below.



```
0000 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0010 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0020 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0030 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0040 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0050 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0060 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0070 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0080 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
0090 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
00A0 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
00B0 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
00C0 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
00D0 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
00E0 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF
00F0 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF

0000 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0010 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
0030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

Osc: RC | WDT: On | PWRT: Off | PMWE: On | BOR: On | LVP: On | CP: Off | CPD: Off | 0000
```

Menu bar, button bar and status bar

At the top of the window you’ll find the button bar for the various commands.

File allows *.hex files to be opened, saved, etc. Device opens a list of commands for programming, reading, testing and erasing PICs. Using **Select/Device** you are able to choose the PIC type to be programmed. Furthermore, **Select/Fuse Word** allows the famous PIC configuration byte to be defined. Finally, there’s **Device/IS Loc** which opens the Editor for entering ID byte values.

Options/Select COM has the obvious function of letting you choose a free COM: port on your PC. The first time the software is used, it will automatically select the first free COM: port found on the system, and then save the setting in the ‘Update.ini’ file. If problems occur with the selection of the port, ‘Update.ini’ may be edited manually (for example, using Notepad) to select the port you want. The file may be found in the program directory.

Program memory

The large window shows the program memory contents using hexadecimal notation. The value at each address location may be changed using an editor, which is opened by putting the cursor on a location and double-clicking or pressing the [Enter] key. The editor then allows you to enter the new hexadecimal value.

The start addresses of the rows are shown in the grey boxes at the left-hand side. If the cursor is placed on a certain location, the associated address is displayed (in hexadecimal) at the far right in the status bar.

The scroll bar at the right-hand side of the window allows you to scroll through the entire program memory.

Data memory

The second, somewhat smaller, window displays the data (EEPROM) memory, if available in the selected PIC chip. With EEPROM-less PICs, this area will be inactive. The use of this window is identical to that of the main window discussed above.

Programming socket

Right beside the small window you’ll see an image of the ZIF programming socket with the selected PIC inserted. This is intended to help you avoid wrong insertions, and obviates an ugly and possibly confusing decal on the case.

Working without hardware

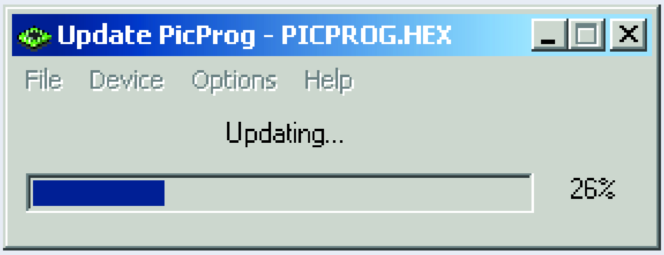
Without the programmer connected up, a window will pop up within a few seconds after starting the program, telling you: ‘No programmer connected. Please connect the programmer and run the program again’. If you press ‘OK’ the program will automatically quit. By pressing ‘Cancel’ the program may be used as a simple hex editor, without access to the program and read functions. Also, a different COM: port may be selected.

Firmware Update utility

A separate program is available for updates to the programmer board firmware (stored in IC6). An update entails downloading new object code in the firmware PIC, IC6. The update mode is selected by fitting jumper JP1. The red LED will light. Once the update is completed, the LED will go out again. If the LED starts to flash, an error has occurred in the download process, which should be repeated until the LED goes out. During the update operation, the program shows the state of the data transfer.

The menu bar contains roughly the same functions as the main program. Most menu options are self evident. Using **Device/Update** you launch an update after having loaded a file (otherwise, the function is not available). If JP1 is not fitted on the programmer board, a message will pop up requesting you to do so.

Once the new code has been successfully copied into IC6, the system will display ‘Update Done’, whereupon the jumper may be removed and the programmer board switched off and on again to use the PicProg 2003 with its new firmware.



Updating...

26%

far enough from the case, its height may be adjusted by stacking up extra socket strips.

Software

The software developed for the PICPro 2003 programming system comes in two parts: the executable code (object code) for IC6 and the Windows software to control the programmer using your PC. The code for IC6 contains a number of routines that look after the communication between IC6 and the PIC device to be programmed. There are also complete read and program algorithms for the supported PIC devices. The code can be updated any time.

Finally, the last 256 bytes of code contain the RS232 routines and the setup routines of the various I/O pins and registers, as well as a routine that allows the remaining part of the program memory to be updated. That particular part is protected to prevent it accidentally being written to, although the update routine only allows valid addresses (valid meaning from 0x0004 to 0x0EFF).

The Windows software consists of two utilities: 'PicProg' which looks after all programming tasks and 'Update PicProg' for updating the object code (firmware) in IC6. The functionality of these two programs is discussed in the inset. The author plans to supply frequent updates of the programs as well as the hex code to ensure the PICProg 2003 project remains 'aware' of the latest developments in PIC land. If and when available, updates will be posted on the *Elektor Electronics* website.

(010102-1)

Advertisement